Acquiring Configuration Knowledge Bases in the Semantic Web using UML

Alexander Felfernig[1], Gerhard Friedrich[1], Dietmar Jannach[1],
Markus Stumptner[2], and Markus Zanker[1]

[1]Institut für Wirtschaftsinformatik und Anwendungssysteme, Produktionsinformatik,
Universitätsstrasse 65-67, A-9020 Klagenfurt, Austria,
email: {felfernig,friedrich,jannach,zanker}@ifit.uni-klu.ac.at.
[2] University of South Australia, Advanced Computing Research Centre,
5095 Mawson Lakes (Adelaide), SA, Australia
email: mst@cs.unisa.edu.au.

Abstract. The Semantic Web will provide the conceptual infrastructure to allow new forms of business application integration. This paper outlines our approach for integrating Web-based sales systems for highly complex customizable products and services (configuration systems) making use of descriptive representation formalisms of the Semantic Web. The evolving trend towards highly specialized solution providers cooperatively offering configurable products and services to their customers requires the extension of current (standalone) configuration technology with capabilities of knowledge sharing and distributed configuration problem solving. On the one hand, a standardized representation language is needed in order to tackle the challenges imposed by heterogeneous representation formalisms of state-of-the-art configuration environments (e.g. description logic or predicate logic based configurators), on the other hand it is important to integrate the development and maintenance of configuration systems into industrial software development processes. We show how to support both goals by demonstrating the applicability of the Unified Modeling Language (UML) for configuration knowledge acquisition and by providing a set of rules for transforming UML models into configuration knowledge bases specified by languages such as OIL or DAML+OIL which represent the foundation for potential future description standards for Web services.

1 Introduction

There is an increasing demand for applications providing solutions for configuration tasks in various domains (e.g. telecommunications industry, automotive industry, or financial services) resulting in a set of corresponding configurator implementations (e.g. [2,11,13,22]). Informally, configuration can be seen as a special kind of design activity [16], where the configured product is built from a predefined set of component types and attributes, which are composed conforming to a set of corresponding constraints.

Triggered by the trend towards highly specialized solution providers cooperatively offering configurable products and services, joint configuration by a set of business partners is becoming a key application of knowledge-based configuration systems. The configuration of virtual private networks (VPNs) [9] or the configuration of enterprise

network solutions are application examples for distributed configuration processes. In the EC-funded research project CAWICOMS[1] the paradigm of Web services is adopted to accomplish this form of business application integration [8]. In order to realize a dynamic matchmaking between service requestors and service providers, configuration services are represented as Web services describing the capabilities of potentially cooperating configuration systems. Currently developed declarative languages (e.g., DAML-S[2]) for semantically describing the capabilities of a Web-service are based on DAML+OIL, that is why we show how the concepts needed for describing configuration knowledge can be represented using semantic markup languages such as OIL [10] or DAML+OIL [20].
The Unified Modeling Language (UML) [15] is a widely adopted modeling language in industrial software development. Based on our experience in building configuration knowledge bases using UML [5], we show how to effectively support the construction of Semantic Web configuration knowledge bases using UML as a knowledge acquisition frontend. The approach presented in this paper enhances the application of Software Engineering techniques to knowledge-based systems by providing a UML-based knowledge acquisition frontend for configuration systems. Vice versa, reasoning support for Semantic Web ontology languages can be exploited for checking the consistency of UML configuration models. The resulting configuration knowledge bases enable knowledge interchange between heterogenous configuration environments as well as distributed configuration problem solving in different supply chain settings.
The paper is organized as follows. In Section 2 we discuss the representative concepts for configuration knowledge bases and in Section 3 we give a description logic based definition of a configuration task as basis for the translation of UML configuration models into a corresponding OIL-based representation.

2 Configuration knowledge representation

Knowledge-based configuration systems build on a configuration model, that represents the generic product structure. The representations concepts for modeling generic product structures are defined in the de facto standard configuration ontologies [5,18] that are based on Ontolingua [12] and represent a synthesis of resource-based [13], function-based, connection-based [14], and structure-based [19] configuration approaches:

- **Component types.** Component types represent the basic building blocks a final product can be built of. They are characterized by attributes.
- **Generalization hierarchies.** Component types with a similar structure are arranged in generalization hierarchies.
- **Part-whole relationships.** Part-whole relationships between component types state the range of subparts an aggregate consists of.
- **Compatibilities and requirements.** Some types of components must not be used together within the same configuration, i.e. they are incompatible. In other cases,

[1] CAWICOMS is the acronym for Customer-Adaptive Web Interface for the Configuration of products and services with Multiple Suppliers (EC-funded project IST-1999-10688).
[2] See http://www.daml.org/services for reference.

the existence of one component of a specific type requires the existence of another specific component within the configuration.
- **Resource constraints.** Parts of a configuration task can be seen as a resource balancing task, where some of the component types produce some resources and others are consumers.
- **Port connections.** In some cases the product topology - i.e., exactly how the components are interconnected - is of interest in the final configuration. The concept of a *port* is used for this purpose.
- **Constraints.** The basic structure of the product is modeled using the aforementioned modeling concepts. In addition, constraints which are related to technical restrictions and economic factors can be expressed on the product model.

In the Knowledge Acquisition Workbench of the CAWICOMS Project graphical representation concepts of the Unified Modeling Language (UML) [15] are used to allow the domain expert acquiring and maintaining the configuration models. In order to allow the refinement of the basic meta-model with domain-specific modeling concepts, UML provides the concept of *profiles* - the configuration domain specific modeling concepts are the constituting elements of a UML *configuration profile* which can be used for building configuration models.
UML profiles can be compared with ontologies discussed in the AI literature. UML *stereotypes* are used to further classify UML meta-model elements (e.g. classes, associations, dependencies). Stereotypes are the basic means to define domain-specific modeling concepts for profiles (e.g. for the configuration profile).

3 Translation of UML configuration models into OIL

In the following we give a description logic based definition of a configuration task [6] and present some example rules to automatically translate UML configuration models into a corresponding OIL representation. The definition is based on a schema S=($\mathcal{CN}$, $\mathcal{RN}$, $\mathcal{IN}$) of disjoint sets of names for concepts, roles, and individuals [3], where $\mathcal{RN}$ is a disjunctive union of roles and features.

Definition 1 (Configuration task): In general we assume a configuration task is described by a triple (DD, SRS, $CLANG$). DD represents the domain description of the configurable product and SRS specifies the particular system requirements defining an individual configuration task instance. $CLANG$ comprises a set of concepts $C_{Config} \subseteq \mathcal{CN}$ and a set of roles $R_{Config} \subseteq \mathcal{RN}$ which serve as a configuration language for the description of actual configurations. A configuration knowledge base KB = $DD \cup SRS$ is constituted of sentences in a description language. □

In addition we require that roles in $CLANG$ are defined over the domains given in C_{Config}, i.e. $range(R_i)$ = $CDom$ and $dom(R_i)$ = $CDom$ must hold for each role $R_i \in R_{Config}$, where $CDom \doteq \bigsqcup_{C_i \in C_{config}} C_i$. We impose this restriction in order to assure that a configuration result only contains individuals and relations with corresponding definitions in C_{Config} and R_{Config}.

Based on this definition, a corresponding configuration result (solution) is defined as follows [6], where the semantics of description terms are given using an interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, (\cdot)^{\mathcal{I}} \rangle$, where $\Delta^{\mathcal{I}}$ is a domain of values and $(\cdot)^{\mathcal{I}}$ is a mapping from concept descriptions to subsets of $\Delta^{\mathcal{I}}$ and from role descriptions to sets of 2-tuples over $\Delta^{\mathcal{I}}$.

Definition 2 (Valid configuration): Let $\mathcal{I} = \langle \Delta^{\mathcal{I}}, (\cdot)^{\mathcal{I}} \rangle$ be a model of a configuration knowledge base KB, $CLANG = C_{config} \cup R_{config}$ a configuration language, and $CONF = COMPS \cup ROLES$ a description of a configuration. $COMPS$ is a set of tuples $\langle C_i, INDIVS_{C_i} \rangle$ for every $C_i \in C_{config}$, where $INDIVS_{C_i} = \{ci_1, \ldots, ci_{n_i}\} = C_i^{\mathcal{I}}$ is the set of individuals of concept C_i. These individuals identify components in an actual configuration. $ROLES$ is a set of tuples $\langle R_j, TUPLES_{R_j} \rangle$ for every $R_j \in R_{config}$ where $TUPLES_{R_j} = \{\langle rj_1, sj_1 \rangle, \ldots, \langle rj_{m_j}, sj_{m_j} \rangle\} = R_j^{\mathcal{I}}$ is the set of tuples of role R_j defining the relation of components in an actual configuration.□

The automatic derivation of an OIL-based configuration knowledge base requires a clear definition of the semantics of the used UML modeling concepts. The semantics of UML configuration models are given by a set of corresponding translation rules. The resulting knowledge base restricts the set of possible configurations, i.e. enumerates the possible instance models which strictly correspond to the UML class diagram defining the product structure. For obvious space restrictions only the translation rule for part-whole relationships is shown:

Part-whole relationships are important model properties in the configuration domain. In [1,17,18] it is pointed out that part-whole relationships have quite variable semantics depending on the regarded application domain. In most configuration environments, a part-whole relationship is described by the two basic roles *partof* and *haspart*. In the following these two basic roles are introduced. Multiplicities used to describe a part-whole relationship denote how many parts the aggregate can consist of and between how many aggregates a part can be shared if the aggregation is non-composite.

Rule (Part-whole relationships): Let w and p be component types in a graphical UML representation, where p is a part of w and ub_p is the upper bound, lb_p the lower bound of the multiplicity of the part, and ub_w is the upper bound, lb_w the lower bound of the multiplicity of the whole. Furthermore let *w-of-p* and *p-of-w* denote the names of the roles of the part-whole relationship between w and p, where *w-of-p* denotes the role connecting the part with the whole and *p-of-w* denotes the role connecting the whole with the part, i.e., *p-of-w* $\sqsubseteq haspart$, *w-of-p* $\sqsubseteq Partof_{mode}$, where $Partof_{mode} \in \{partof_{composite}, partof_{shared}\}$. The roles $partof_{composite}$ and $partof_{shared}$ are assumed to be disjoint, where $partof_{composite} \sqsubseteq partof$ and $partof_{shared} \sqsubseteq partof$. DD is extended with

class-def p.

class-def w.

slot-def *w-of-p* subslot-of $Partof_{mode}$ inverse *p-of-w* domain p range w.

slot-def *p-of-w* subslot-of *haspart* inverse *w-of-p* domain w range p.

p: slot-constraint *w-of-p* min-cardinality lb_w w.

p: slot-constraint *w-of-p* max-cardinality ub_w w.

w: slot-constraint *p-of-w* min-cardinality $lb_p\ p$.
w: slot-constraint *p-of-w* max-cardinality $ub_p\ p$. □

Remark: The semantics of *shared* part-whole relationships ($partof_{shared} \sqsubseteq partof$) are defined by simply restricting the upper bound and the lower bound of the corresponding roles. In addition the following restriction must hold for each concept using partof relationships:

(((*slot-constraint* $partof_{composite}$ *cardinality 1 top*) *and* (*slot-constraint* $partof_{shared}$ *cardinality 0 top*)) *or* (*slot-constraint* $partof_{composite}$ *cardinality 0 top*)).

This restriction denotes the fact that a component which is connected to a whole via composite relationship must not be connected to any other component. □

For further details, an example and the complete set of translation rules see the long version of this paper [7].

4 Conclusions

The application of the modeling concepts presented in this paper has its limits when building configuration knowledge bases - in some domains there exist complex constraints that do not have an intuitive graphical representation. Happily, (with some minor restrictions discussed in [6]) we are able to represent such constraints using languages such as OIL or DAML+OIL. UML itself has an integrated constraint language (Object Constraint Language - OCL [21]) which allows the formulation of constraints on object structures. The translation of OCL constraints into representations of Semantic Web ontology languages is the subject of future work, a translation into a predicate logic based representation of a configuration problem has already been discussed in [4]. The current version of our prototype workbench supports the generation of OIL-based configuration knowledge bases from UML models which are built using the modeling concepts presented in this paper, i.e. concepts for designing the product structure and concepts for defining basic constraints (e.g. *requires*) on the product structure.

References

1. A. Artale, E. Franconi, N. Guarino, and L. Pazzi. Part-Whole Relations in Object-Centered Systems: An Overview. *Data & Knowledge Engineering*, 20(3):347–383, 1996.
2. V.E. Barker, D.E. O'Connor, J.D. Bachant, and E. Soloway. Expert systems for configuration at Digital: XCON and beyond. *Communications of the ACM*, 32(3):298–318, 1989.
3. A. Borgida. On the relative expressive power of description logics and predicate calculus. *Artificial Intelligence*, 82:353–367, 1996.
4. A. Felfernig, G. Friedrich, and D. Jannach. Generating product configuration knowledge bases from precise domain extended UML models. In *Proceedings of the* 12^{th} *International Conference on Software Engineering and Knowledge Engineering (SEKE'2000)*, pages 284–293, Chicago, USA, 2000.
5. A. Felfernig, G. Friedrich, and D. Jannach. UML as domain specific language for the construction of knowledge-based configuration systems. *International Journal of Software Engineering and Knowledge Engineering (IJSEKE)*, 10(4):449–469, 2000.

6. A. Felfernig, G. Friedrich, D. Jannach, M. Stumptner, and M. Zanker. A Joint Foundation for Configuration in the Semantic Web. *Proceedings of the Workshop on Configuration (ECAI'2002)*, 2001.
7. A. Felfernig, G. Friedrich, D. Jannach, M. Stumptner, and M. Zanker. Transforming UML domain descriptions into Configuration Knowledge Bases for the Semantic Web. Lyon, France, 2002.
8. A. Felfernig, G. Friedrich, D. Jannach, and M. Zanker. Semantic Configuration Web Services in the CAWICOMS Project. Sardinia, Italy, 2002.
9. A. Felfernig, G. Friedrich, D. Jannach, and M. Zanker. Web-based Configuration of Virtual Private Networks with Multiple Suppliers. Cambridge, UK, 2002. Kluwer Academic Publisher.
10. D. Fensel, F. vanHarmelen, I. Horrocks, D. McGuinness, and P.F. Patel-Schneider. OIL: An Ontology Infrastructure for the Semantic Web. *IEEE Intelligent Systems*, 16(2):38–45, 2001.
11. G. Fleischanderl, G. Friedrich, A. Haselböck, H. Schreiner, and M. Stumptner. Configuring Large Systems Using Generative Constraint Satisfaction. *IEEE Intelligent Systems*, 13(4):59–68, 1998.
12. T. Gruber. Ontolingua: A mechanism to support portable ontologies. *Technical Report KSL 91-66*, 1992.
13. E.W. Jüngst M. Heinrich. A resource-based paradigm for the configuring of technical systems from modular components. In *Proceedings of the* 7^{th} *IEEE Conference on AI applciations (CAIA)*, pages 257–264, Miami, FL, USA, 1991.
14. S. Mittal and F. Frayman. Towards a Generic Model of Configuration Tasks. In *Proceedings* 11^{th} *International Joint Conf. on Artificial Intelligence*, pages 1395–1401, Detroit, MI, 1989.
15. J. Rumbaugh, I. Jacobson, and G. Booch. *The Unified Modeling Language Reference Manual*. Addison-Wesley, 1998.
16. D. Sabin and R. Weigel. Product Configuration Frameworks - A Survey. In B. Faltings and E. Freuder, editors, *IEEE Intelligent Systems, Special Issue on Configuration*, volume 13, pages 50–58. IEEE, 1998.
17. U. Sattler. Description Logics for the Representation of Aggregated Objects. In *Proceedings of the* 14^{th} *European Conference on Artificial Intelligence (ECAI 2000)*, pages 239–243, Berlin, Germany, 2000.
18. T. Soininen, J. Tiihonen, T. Männistö, and R. Sulonen. Towards a General Ontology of Configuration. *AI Engineering Design Analysis and Manufacturing Journal, Special Issue: Configuration Design*, 12(4):357–372, 1998.
19. M. Stumptner. An overview of knowledge-based configuration. *AI Communications*, 10(2), June, 1997.
20. F. vanHarmelen, P.F. Patel-Schneider, and I. Horrocks. A Model-Theoretic Semantics for DAML+OIL. *www.daml.org*, March 2001.
21. J. Warmer and A. Kleppe. *The Object Constraint Language - Precise Modeling with UML*. Addison Wesley Object Technology Series, 1999.
22. J.R. Wright, E. Weixelbaum, G.T. Vesonder, K.E. Brown, S.R. Palmer, J.I. Berman, and H.H. Moore. A Knowledge-Based Configurator that supports Sales, Engineering, and Manufacturing at AT&T Network Systems. *AI Magazine*, 14(3):69–80, 1993.